Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 58 (2015) 257 – 264

Procedia Computer Science

Second International Symposium on Computer Vision and the Internet(VisionNet'15)

Automatic Answer Assessment in LMS using Latent Semantic Analysis

Thomas N T[a], Ashwini Kumar[b], Kamal Bijlani[c]

[a]*Amrita E-leaning Technologies, Amrita Vishwa Vidyapeetham, Amritapuri, Kollam, India*
thomasnt13@gmail.com
[b,c]*Amrita E-learningResearch Lab, Amrita Vishwa Vidhyapeetham, Amritapuri, Kollam, India*
{ashwinikumar,kamal}@am.amrita.edu}

Abstract

E-Learning uses systems like Learning Management Systems (LMSs), to support and enhance teaching-learning process. When we compared some of the popular LMSs, a lack of efficient mechanisms to assess descriptive answers like long answers and essays, was observed. Also, it is tedious for the teachers to manually evaluate for large number of students. Hence, we propose an automatic system to assess descriptive answers of students and provide teachers with immediate feedback. This is achieved by first, comparing student's answer with teacher's ideal answer set, using latent semantic analysis (LSA). Then, estimating the order of previous and upcoming words in an answer using positional indexing, based on the keyword list added by the teacher. A final score is then generated depending on LSA, correct keyword usage and also on spell check. Cohen's kappa coefficient of human rater-tool agreement showed a good strength when the system was integrating into an existing LMS.

Peer-review under responsibility of organizing committee of the Second International Symposium on Computer Vision and the Internet (VisionNet'15)

Keywords: E-learning,Latent Semantic Analysis, Learning Management Systems, Positional Indexing;

1. Introduction

E-Learning makes use of the software tools to help instructors and students. The availability of different tools in Learning Management System (LMS) helps instructors to teach their courses. Also, it can offer a great deal of flexibility in several aspects such as when, where and how educational resources are distributed[1]. Learning Management System is a multi-user software for managing administration, content delivery and student tracking. The actual LMS users are instructors and learners. Instructors create and manage class schedules, resources,

Peer-review under responsibility of organizing committee of the Second International Symposium on Computer Vision and the Internet (VisionNet'15)
doi:10.1016/j.procs.2015.08.019

attendance, track scores, progress and other learning activities etc. LMS learners register for classes, search courses, receive notifications, earn credits etc. Some widely known LMSs are Moodle, Canvas, Blackboard, Sakai, eCollege, Desire2Learn etc. These systems provide almost all core features of LMSs. There are open source and commercial LMSs.

Assessment of student performance is important as that would help the teachers understand the level of each student. There are many plugins in LMS that provide this functionality. These plugins make submission of assignment, records of marks and feedback process easy and efficient for the users. However there is a limited availability of such plugins or tools in automatically grading question types for long and descriptive answers. Typically, there are many students enrolled to a course. In such a scenario, it is nearly impossible for an instructor to go through each answer to evaluate their answer.

As a solution to this problem, the paper proposes an automatic assessment model to assess the answers of students and provide the teachers with immediate feedback. Latent Semantic Analysis (LSA) is used in this paper for automatic assessment of student answers. It is a method used in natural language processing to examine the connection between set of documents and terms. A word by document matrix is generated, where each rows represents terms and column represents documents. Singular value decomposition is used to convert a large piece of text into a reduced matrix by preserving the similarity among columns and reducing the number of rows. The similarity between terms are compared by taking cosine angle between the two row vectors. The cosine similarity with value 1 denotes similar terms and values near 0 represents dissimilar terms. The system then checks the order of the keywords written by students using positional indexing. The system also detects the common spelling mistakes in the student submission.

2. Learning Management Systems and assessment capabilities

2.1. Moodle

It is an open-source software platform used in several learning environments. It helps educators to create online courses and is used in many training and development setups.

- *Benefits*: Open source, customizability, large number of plugins, mobile friendliness etc.
- *Limitations:* The essay type questions and long answer questions are graded manually.
- The Moodle *assessment* methods include : Essay format questions, Online audio recording, Multiple choice questions, Short answer questions, Numerical questions, Matching questions, Embedded answers, Random short-answer, Pdf upload option, Interactive video and audio recording.

2.2. Blackboard Learn

Blackboard Learn is a student centred LMS that enhances basic teaching tasks. It offers a free version for evaluation. It provides customizable course management and delivery. It also provides intuitive online structure for instructors to organize courses.

- The Blackboard Learn assessment methods include: Essay type questions, File upload questions, Fill in the blanks, Hotspot, Matching questions, Ordering questions, Jumbled sentences, Multiple choice, Quiz bowl, Short answer type questions, True/False questions, Random block, Form question pool.
- *Benefits*: Simple interface and easy to navigate, encourage student interaction, easy management of learning content, simple file management system, personal E-Portfolios, instant collaboration, mobile friendliness etc.
- *Limitations*: Lack of notifications, Self and peer assessment facilities are available, but evaluator should grade descriptive answers including essay type questions manually.

2.3. Desire2Learn

Desire2Learn is an institutional LMS. It is a learning environment for online and blended courses. This is mainly used for circulation of handouts, lecture notes, grade updates etc. The underlying technology is Microsoft.Net. It provides an organized interface and spreadsheet view in grades.

- The *assessment* question types in Desire2Learn are Image questions, Long answer questions, Short answer questions, Matching questions, multiple choice, True/False, Multi-select, Ordering type, Arithmetic questions and Significant figures.
- *Benefits*: User friendly interface, accessibility, collaborative environment, flexibility etc.

- *Limitations:* Notification problems, Difficult to upload and download grades to excel, Manual assessment of descriptive answers.

2.4. Sakai

Sakai is mainly used in educational institutions for teaching, research and collaboration. It is also used by government, military and corporates. The underlying technology is java.

- The *assessment* methods in Sakai are Open ended questions, Timed assessments, Audio recording, Matching questions, Quizzes, Multiple choice, Short answer, Essay type questions.
- *Benefits*: Scalable, interoperable, minimize workload, open source, attractive interface, customizable.
- *Limitations:* Essay type questions and descriptive answers are manually assessed by teachers.

3. Related Work

Mingqing Zhang et al.[2] proposed a new incremental technique of latent semantic analysis to score essays when the dataset is huge. The incremental LSA technique has advantage over traditional method in case of memory usage and running time.

Prema Nedungadi and Harsha Raj[3] included an unsupervised word sense disambiguation (WSD) algorithm to an existing automatic essay scoring model to properly recognize the meaning of terms used in a sentence. The paper concludes that model with WSD perform better than model without WSD.

Zuhoor A. Al-Khanjari & Yusra M. Al-Roshdi[4] proposed an idea to overcome the limitations of learning management systems towards the practical aspects of computer science education. The authors have selected Moodle for this experimentation and have developed a plugin software tool called virtual programming lab (VPL). This software helps learners to gain practical knowledge. The software compiles programming languages and to submit assignments.

F.Huertas & A Navarro[5] proposed some integration approaches of the tools used in e-Learning platforms by inclusion of Application Programming Interfaces (API) to make the functionality of the e-Learning platform in the same language as the platform has been built.

4. Applied Method

The sequence diagram in Fig. 1 shows a typical flow of submission and evaluation. The teacher adds a question for evaluation and specifies the time limit. He can also add ideal answers or pre-graded answers to generate a training answer set. The teacher can also add important keywords that he/she needs in student's answer in the order it should appear. He then releases the questionnaire. At this point, the students in the classroom receives the questionnaire, as shown in Fig. 2(a), and can start answering. They can then write the answers in the input console and submit, respecting the time limit.

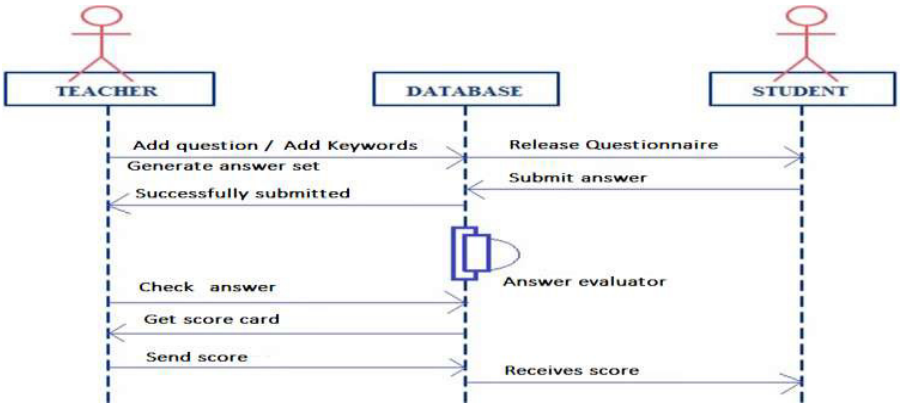

Fig 1: Teacher student interaction diagram

The instructor gets notification at the same time when the student submits their work. If the instructor side doesn't receive any answer from a particular student, then an automatic notification will be sent to the student. The submitted answers are then checked by the automatic answer evaluator. The automatic answer evaluator checks the submitted answer and generates a scorecard for the teacher. The teacher then sends the score card to the students.

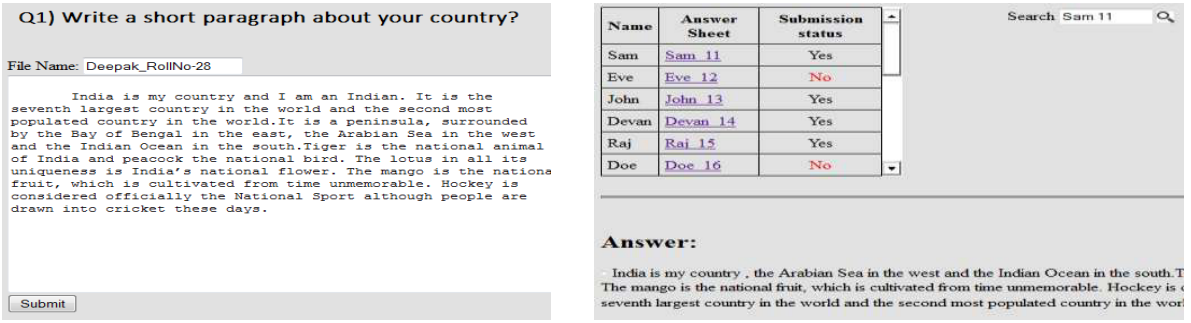

Fig 2. (a)Student Module (b) Instructor module

The instructor module is shown in Fig. 2(b). This view shows the list of all students who participated in test. A search option is there to search a particular student; and the teacher can follow the link to read the submitted answer.

5. Proposed Tool

The system assesses the descriptive answers of the students by comparing that with the ideal answer set provided by the teacher and provides the teachers with a score almost instantly. Fig. 3 shows the workflow of the system.

The system consists of three main steps:

1. Latent semantic analysis
2. Positional Indexing
3. Spell check

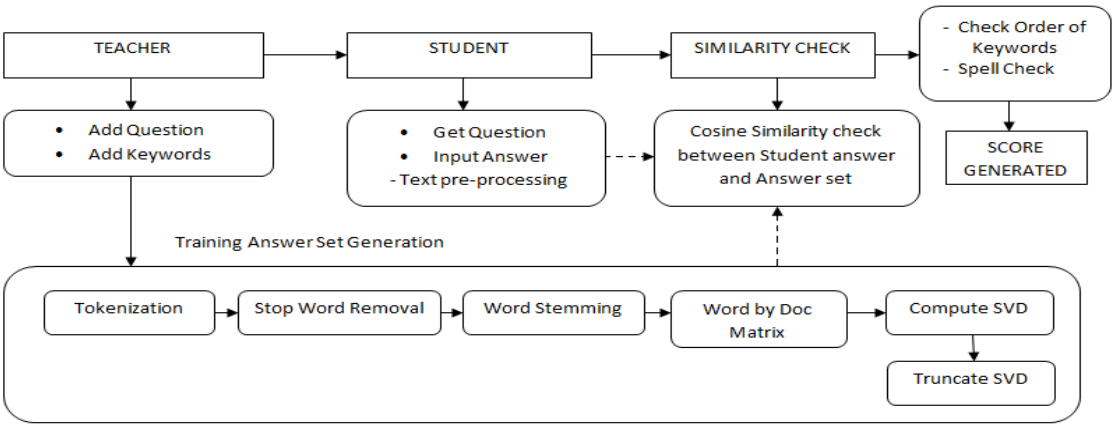

Fig 3: System Workflow

5.1. Latent semantic analysis:

In this step the students work is compared with the anwer set provided by the teacher. This is done by following the steps below:

1. *Tokenization* : The answers are segmented into sentences. The sentences are then divided into words. The

words are treated as tokens.

2. *Stopword removal:*The most common words are filtered out from the text. For example : a, an, an, and, any, are, as, at, be, so, to, is etc.
3. *Stemming of Words*: The words in the answer or document sets are reduced to their root form based on Porter Stemmer algorithm rules given below.
   a. The algorithm discards plurals and 'ed' or 'ing' suffixes.
   b. When there is another vowel in the stem, it changes 'y' to 'i'.
   c. Double suffixes are changed to single ones, as in, -ization, -ational etc.
   d. It deals words with suffixes like -full, -ness etc.
   e. The algorithm takes off -ant, -ence etc. For example, 'Precedent' will be changed into 'preced'
   f. Removes a final -e in words. For example, 'controllable' will be changed into control.
4. *Word by Document Matrix*($S_{t\times d}$): In this step a word by document matrix is formed. Each row of the matrix represents terms in an answer and each column represents documents or answers. The presence of word in a document is represented as 1 and absence as 0 in each cell.
5. *Singular Value Decomposition* : Singular value decomposition helps in factorizing a complex matrix. The word by document matrix is decomposed into the product of 3 matrices (U,Σ, V^T).

$$S_{t\times d} = U_{t\times m} \times \Sigma_{m\times m} \times V^T_{d\times m} \quad (1)$$

where,

$U_{t\times m}$ matrix consists of one row vector for each term.

$\Sigma_{m\times m}$ is diagonal matrix with singular values.

$V^T_{d\times m}$ matrix consists of one column vector for each term.

The algorithm for SVD is given below:

Algorithm.1-Singular Value Decomposition[8]

1. *Set* $Y = S*S^T$
2. *Calculate* $\lambda_1, \lambda_2, \lambda_3, \ldots \lambda_m$ *Eigenvalues of Y*
3. **for** *i= 1 to n*
4. **do** μ_i= sqrt (λ_i)
5. *end* **for**
6. *Sort* $\mu_1, \mu_2, \mu_3, \ldots \mu_n$ *in descending order*
7. *Initialize* Σ
8. **for** *i= 0 to m*
9. **for** *j= 0 to n*
10. **do if** *i=j*
11. then *set* $\Sigma_{ij}=\mu_i$
12. else then *set* $\Sigma_{ij}=0$
13. *end* **if**
14. *end* **for**
15. *end* **for**
16. **for** *i= 0 to n*
17. **do** μ_i= eigenvector of (λ_i)
18. *end* **for**
19. *Create a matrix* $V_{p\times m}$ *having the* μ_i *as columns*
20. $V^T_{p\times m}$= *the transpose of* $V_{d\times m}$
21. *Calculate* $U_{p\times m} = S_{m\times m} \times V_{p\times m} \times \Sigma^{-1}_{p\times p}$

6. *Truncate SVD:* The singular values from S matrix is then arranged in decreasing order. The dimensionality reduction of SVD matrix is done by removing lower singular values. The corresponding dimensions of columns and rows in matrices U and V are also reduced. The dimensionality reduction algorithm of SVD matrix is shown below.

Algorithm.2-Dimentionality Reduction Algorithm of SVD[8]

1. *Set k = 0*
2. **for** *i= 0 to p-1*
3. **do if** *(Σ_{ii}<0.5)*
4. *then k= i - 1*
7. *end* **if**
8. *Increment i*
9. *end* **for**
10. Σ_k = submatrix of $\Sigma_{p\times p}$of order $k \times k$
11. U_k = submatrix of $U_{m\times p}$ of order $m \times k$
12. V_k^T= submatrix of $V_{p\times n}^T$ of order $k \times$ p
13. *Truncate U,, V^T and make* $S_{k\times k} = U_{m\times k} \times \Sigma_{k\times k} \times V_{k\times p}$
14. **for** *j= 1 to p*
14. **do** *make* answer vector $D_j = D_j^T \times U_{m\times k} \times \Sigma_{k\times k}^{-1}$
15. *end* **for**

The similarity is checked between the submitted answer and the ideal answer vector. To start with the similarity check, first the submitted answer is made to go through test pre-processing steps such as removing stop words and word stemming. Then a word by document matrix is built. A query vector *q,* that is the submitted answer vector, is then generated. By using the formula of cosine similarity[6],

$$\cos\theta = \frac{\sum_{i=1}^{n} q_i \times D_i}{\sqrt{\sum_{i=1}^{n}(q_i)^2} \times \sqrt{\sum_{i=1}^{n}(D_i)^2}} \quad (2)$$

where, *q* being the submitted answer or query vector and *D* the ideal answer vector, we will get the measure of similarity between two vectors *q* and *D*.

5.2. Positional Indexing:

The teachers are given the provision to add the keywords that is relevant to the answer they are seeking. These words are indexed based on their position, to get the order of occurrence of each word. Based on this indexing, the order of previous and upcoming words are checked in the answer and a score is generated.

The order of keywords written by students is estimated using positional indexing. For example, water vapor and vapor water will not give same result. We store the positions of each term. We check for the position of *water* and *vapor* in the tokenized answer. We look for possible match in the position of appearance of each words based on the order of keywords by teacher.

For example:

Water :< answer 1, 4: < 15, 28, 41, 62 >

Vapor: < answer 1, 7: < 6, 13, 23, 29, 47, 63, 113 >

The term water occurs 4 times in answer 1 at positions 15, 18...etc. The term vapor occurs 7 times in answer 1 at positions 6, 13, 23.etc. We will go probable match of term order using positional indexing based on keyword list.

Water: < answer 1: <…, 28, 62…>

Vapor: <answer 1 :<…., 29, 63…>

5.3. Spell Check:

The lists of common misspellings and their correct spellings in English are stored in the database. The list source is Wikipedia [9]. The tool detects common spelling mistakes.

Each mistakes in an answer/document are counted and a final score card is generated. The final score card is based on the latent semantic analysis, correct keyword usage and common misspellings.

6. Experimental Results

We gathered data using two different modes of experimentation to support the idea of the need of automatic answer assessment method in LMSs. First is an online survey and the second is performance evaluation using the proposed model.

6.1. Pre-Survey:

The idea of having a survey was to find out whether the existing methods or tools in LMS support practical assessment. The responses from a survey done in social networking site (Facebook) groups like ex - 6.00x Introduction to Computer Science and Programming, MIT 6.00x Facebook Group, MIT 6Xseries Facebook Group, E-Learning Technologies Group etc. are shown in the Fig. 4. Based on the survey responses, there is a limited availabilty of tools to automatically assess the descriptive answers written by students. Instructors take time to manually check the written answers of large number of students and it is difficult for teachers to give an immediate feedback on assessment.

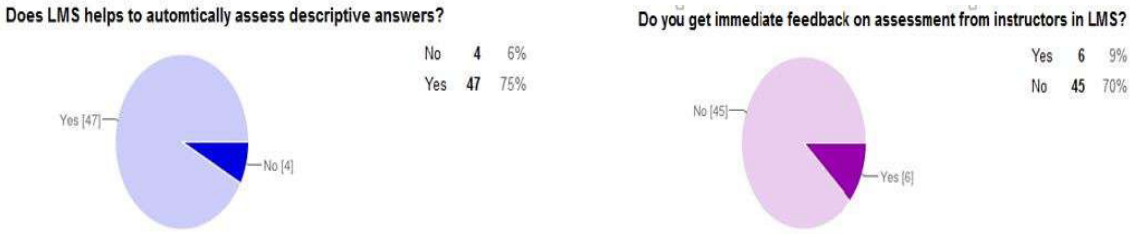

Fig 4: Survey responses from e- learning groups in Facebook

6.2. Performance Evaluation of Tool:

Amrita University has developed an integrated university learning management system called Amrita University Management System (AUMS). The proposed software tool is integrated to AUMS for testing. The proposed tool is tested for written answers submitted by students. We tested our tool using a dataset of 252 answers. The entire experiment is repeated for 3 different sets of students. The answers of each set of students are evaluated by human grader. Then students are categorized on the basis of grades as A, B, C. The percentage of observed agreement in giving grades between human grader and tool is,

$$P(A) = \frac{N_a}{T_s} \quad (3)$$

Where

N_a-Actual number of agreement between model and human rater in giving a grade

T_s-Total no of students.

The probability of expected or chance of agreement,

$$P(B) = (E_a + E_b + E_c) \quad (4)$$

E_a -Expected agreement for A grade.

E_b-Expected agreement for B grade.

E_c-Expected agreement for C grade.

The agreement percentage between the scores by tool and teachers are calculated for each set. The probability of actual agreement P(A) and the probability of expected or occurs by chance P(B) is calculated. The probability and strength of agreement is shown in Fig. 5.

Cohen's Kappa method[7] is used to get the inter-rater agreement. The percentage agreement and kappa score are calculated for each set and shown in table I.

Cohen's Kappa coefficient, $\mathbb{K} = \frac{P(A)-P(B)}{1-P(B)}$ (5)

These results show the strength of agreement between teachers and tool is good (0.40 to 0.75).

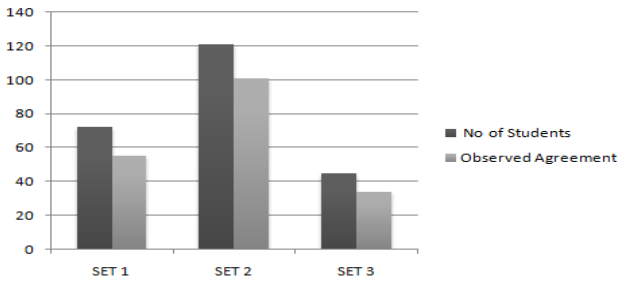

Fig 5: Teacher-tool observed agreement

Table 1. Cohen's kappa score and strength of agreement

Set	Grade A	Grade B	Grade C	Total students	Agreement	Agreement expected by chance	Kappa score	Strength of agreement
1	21	16	18	72	55(76.39%)	24.4(33.85%)	0.643	Good
2	55	25	21	121	101(83.47%)	45.9(37.96%)	0.734	Good
3	12	15	7	45	34(75.56%)	16.1(35.85%)	0.619	Good

Conclusion

This paper has presented an automatic assessment tool that assesses the descriptive answers submitted by students in LMS. This is achieved by using Latent Semantic Analysis (LSA) to compare the submitted answers with ideal answer set. The system then indexes the answer using positional indexing to evaluate the correct usage of words provided by the teacher as keywords. Spell check is also done in the answers by the system. A final score is then generated depending on the results obtained from LSA, correct keyword usage and spell check. This score is provided to the evaluator immediately after the student submits their answers. The Cohen's kappa coefficient of human rater-tool agreement showed a good strength when the system was evaluated after integrating it into an existing LMS. Experimental results show that this method can be effectively used in LMS. In future, the proposed work can be extended to check the syntax of submitted answers in LMS.

References

1. X., Huan, R., Shehane & A., Ali. “Teaching computer science courses in distance learning”. *Journal of Instructional Pedagogies*, 2011, Vol. 6, pp.47-60.
2. Mingqing Zhang, Shudong Hao. “Automated Essay Scoring Using Incremental Latent Semantic Analysis”. *Journal Of Software*, 2014, Vol. 9, No. 2,
3. Prema Nedungadi and Harsha Raj. “Unsupervised Word Sense Disambiguation for Automatic Essay Scoring”. *Advanced Computing, Networking and Informatics* - Volume 1, Smart Innovation, Systems and Technologiespg.27, 2014.
4. Zuhoor A. Al-Khanjari &Yusra M. Al-Roshdi. “Extending the Functionality of LMS to Support Computer Science Education Using Plug-in Tools”. *Journal of Emerging Technologies in Web Intelligence*, VOL. 6, NO. 2, May 2014.
5. F., Huertas & A., Navarro. “Integration Mechanisms in e-learning Platforms”. *International Journal of Computer Information Systems and Industrial Management Applications*. Vol. 5, pp. 714-721, 2013.
6. Cosine similarity: http://en.wikipedia.org/wiki/Cosine_similarity.
7. Cohen's Kappa: http://en.wikipedia.org/wiki/Cohen%27s_kappa.
8. Md. Monjurul Islam, A. S. M. Latiful Hoque."Automated Essay Scoring Using Generalized Latent Semantic Analysis". *Journal of Computers*, Vol. 7, No. 3, 2012.
9. *Wikipedia: Lists of common misspellings/Grammar and Misc.:* http://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings/Grammar_and_Misc.